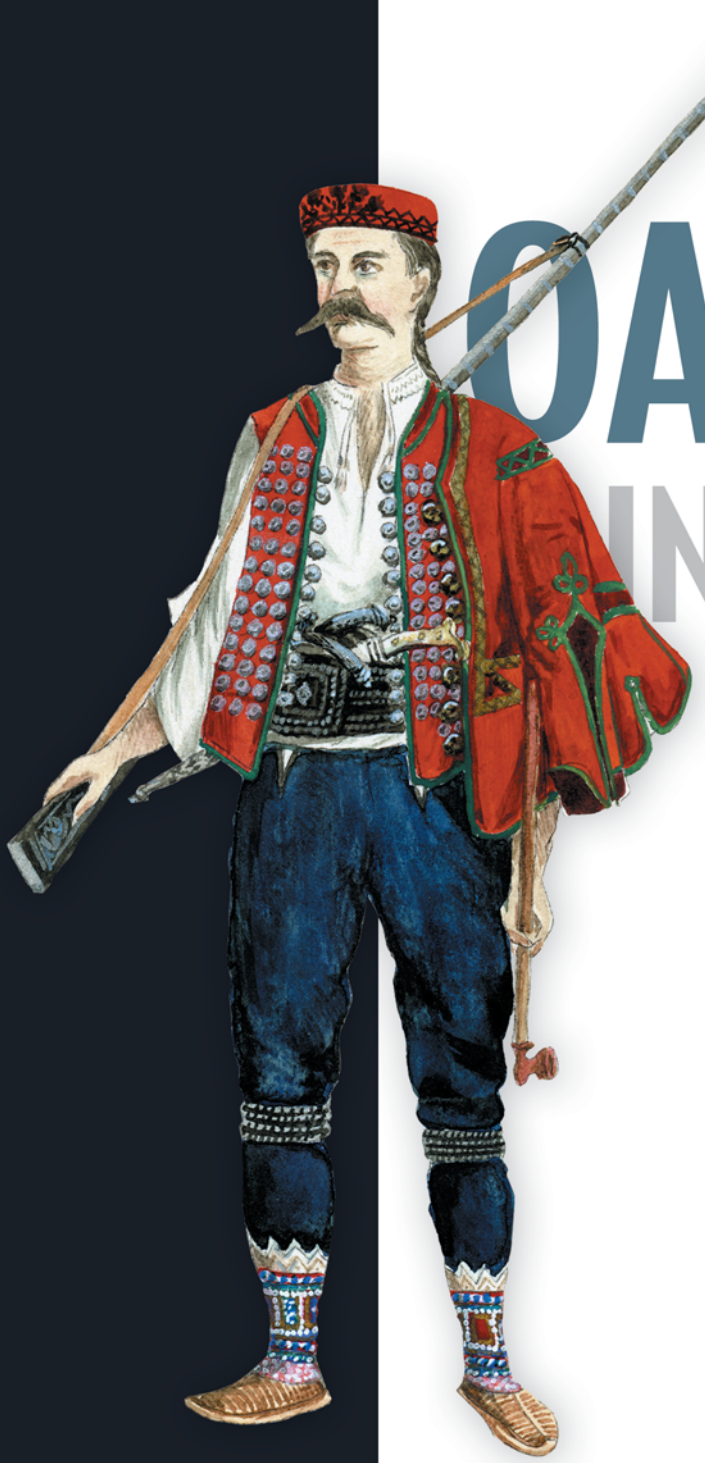


OAuth 2 IN ACTION

Justin Richer
Antonio Sanso

FOREWORD BY Ian Glazer

MANNING

OAuth 2 in Action

JUSTIN RICHER
ANTONIO SANSO

MANNING
SHELTER ISLAND

For online information and ordering of this and other Manning books, please visit www.manning.com. The publisher offers discounts on this book when ordered in quantity. For more information, please contact

Special Sales Department
Manning Publications Co.
20 Baldwin Road
PO Box 261
Shelter Island, NY 11964
Email: orders@manning.com

Manning Publications Co.
20 Baldwin Road
PO Box 261
Shelter Island, NY 11964

Development editor: Jennifer Stout
Technical development editors: Dennis Sellinger
David Fombella Pombal
Copyeditor: Progressive Publishing Services
Technical proofreader: Ivan Kirkpatrick
Composition: Progressive Publishing Services
Cover design: Marija Tudor

ISBN: 9781617293276
Printed in the United States of America
1 2 3 4 5 6 7 8 9 10 – EBM – 22 21 20 19 18 17

brief contents

contents

foreword

There is nothing more daunting than a blank page. It stares at you. It taunts you.

It's not like you don't know what you want to do. You have a clear picture of the awesome that you want to unleash. You can almost envision the smiles on your boss's or customer's face as they delight in the awesome you will create. But the problem is there's a blank page in front of you.

So you reach for your tools. Because you're reading this, it's likely you are a developer or identity management professional. Either way, you know that security is paramount and you want to protect the awesome that you intend to build.

Enter OAuth. You've heard of it. You know it has something to do with protecting resources—most notably APIs. It's super popular and, apparently, it can do anything. And the problem with things that can do *anything* is that they make it hard to do *something*. They are yet another blank page.

Enter Justin and Antonio and this book. The easiest way to get over the paralysis when working with a thing that can do anything is to start and just try to do something. This book not only explains what OAuth does, it gently guides you through the process of doing something, at the end of which not only will you have a very solid understanding of OAuth as a tool, but you'll no longer have a blank page in front of you—you'll be ready to deliver the awesome that's in your head.

OAuth is a very powerful tool. Its power comes from its flexibility. Flexibility often means the ability to not only do what you want to do, but also the ability to do things in an unsafe way. Because OAuth governs access to APIs, which in turn gates access to your important data, it's crucial that you do use it in a safe way by avoiding antipatterns and using best practices. Stated differently, just because you have the flexibility to do anything and deploy in any way, doesn't mean that you should.

There's another thing about OAuth we might as well get on the table—you are not working with OAuth because you want to work with OAuth. You are working with OAuth because you want to do something else—most likely orchestrate a bunch of API calls and then do something awesome with the results. You're thinking about a full page; you're thinking about the awesome you want to unleash. OAuth is a way to get there, and to get there more securely.

Thankfully, Justin and Antonio provide pragmatic guidance on what to do and what not to do. They acknowledge both the "I just want to get this done" and the "I want to make sure this is secure" mindsets you have.

With the page filled, with the awesome out of your head and in your customers' hands, you realize the job wasn't so hard after all.

—IAN GLAZER
SENIOR DIRECTOR, IDENTITY
SALESFORCE

preface

My name is Justin Richer, and I'm not a classically trained security nerd, even though I pretend to be one for my day job as a consultant. My background is in collaboration technologies and how we can get people doing things together using computers. Even so, I've been working with OAuth for a long time, having implemented several early OAuth 1.0 servers and clients to connect the collaboration systems that I was conducting research with at the time. It was around then that I came to appreciate that you needed to have a good, implementable, usable security system if your application architecture was going to survive in the real world. Around this time, I attended the early Internet Identity Workshop meetings, where people were talking about a next generation of OAuth, something that would build on the lessons learned from using OAuth 1.0 out in the real world. When the development of OAuth 2.0 started up in the Internet Engineering Task Force (IETF), I joined the group and dove face first into the debates. Several years later, we came up with a specification. It wasn't perfect, but it worked pretty well, people got it, and it caught fire.

I stayed involved with the OAuth Working Group, and even served as editor for the Dynamic Registration (RFC 7591 and 7592) and Token Introspection (RFC 7662) extensions to OAuth. Today, I'm the editor or author for parts of the OAuth Proof of Possession (PoP) suite, as well as the technical editor for several profiles and extensions of OAuth and its related protocols. I worked on the OpenID Connect core specification, and my team and I implemented a fairly well-received OAuth and OpenID Connect server and client suite, MITREid Connect. I suddenly found myself talking about OAuth 2.0 to many different audiences and implementing it

on a wide variety of systems. I'd taught classes, given lectures, and written a handful of articles about the subject.

So when Antonio Sanso, a well-respected security researcher in his own right, approached me to write this book together, it made sense for me to jump in. We looked around at what books were available on OAuth 2.0, and were unimpressed. Most of the material we found was specific to a service: *How to write an OAuth client to talk to Facebook or Google*, for instance. Or *How to authorize your native application to GitHub's API*. And if that's all you care about, there's plenty of material out there. But what we didn't see was something that would take the reader through the entire OAuth system, explaining why it is designed the way that it is, pointing out its flaws and limitations as well as its strengths. We decided that there was a need for a more comprehensive approach, and we decided to make it the best that we could. Consequently, this book doesn't talk to any specific real-world OAuth provider, nor does it get into detail on a particular API or vertical domain. Instead, this book focuses on doing OAuth for its own sake, so that you can see how all the gears mesh together when you turn the cranks.

We built out a code framework that, we hoped, would allow readers to focus on the core aspects of OAuth without getting overly caught up in the implementation platform details. After all, we didn't want a book that was "How to implement OAuth 2.0 on Platform Du Jour," but rather, "How the nuts and bolts of OAuth 2.0 work so you can use whatever platform you want." So we went with a relatively simple Node.js framework, built on Express.js, and liberally used library code to abstract away the platform-specific weirdness as much as possible. Still, it's JavaScript, so some of that weirdness crept in from time to time, as it would on any platform. But it's our hope that you will be able to apply the methods and themes used here to your chosen language, platform, and architecture.

Speaking of histories, how did we even get here? The story starts in 2006, when several web service companies, including Twitter and Ma.Gnolia, had complementary applications and wanted their users to be able to connect them together. At the time, this type of connection was typically accomplished by asking the user for their credentials on the remote system and sending those credentials to the API. However, the websites in question used a distributed identity technology, OpenID, to facilitate login. As a consequence, there were no usernames or passwords that could be used for the API.

To overcome this, the developers sought to create a protocol that would allow their users to delegate access to the API. They based their new protocol on several proprietary implementations of this same concept, including Google's AuthSub and Yahoo!'s BBAuth. In all of these, a client application is authorized by a user and receives a token that can then be used to access a remote API. These tokens were all issued with a public and private portion, and this protocol used a novel (if in retrospect fragile) cryptographic signing mechanism so that it could be used over non-TLS HTTP connections. They called their protocol OAuth 1.0 and published it as an open standard on the web.

It quickly gained traction, and free implementations in several languages were made available alongside the specification itself. It worked so well and developers liked it so much that even the large internet companies soon deprecated their own proprietary mechanisms that had inspired OAuth in the first place.

As happens with many new security protocols, a flaw was found early on in OAuth 1.0's life, leading to the development of OAuth 1.0a to close a session fixation vulnerability. This version was later codified in the IETF as RFC 5849. At this point, a community was beginning to grow around the OAuth protocol, and new use cases were being developed and implemented. Some of these pushed OAuth into places that it was never meant to be used in, but these off-label OAuth uses worked better than any available alternatives. Still, OAuth 1.0 was a monolithic protocol designed to provide one mechanism to solve all use cases, and it was venturing into uncomfortable territory.

Soon after the publication of RFC 5849, the Web Resource Access Protocol (WRAP) was published. This proposed protocol took the core aspects of the OAuth 1.0a protocol—a client, delegation, and tokens—and expanded them to be used in different ways. WRAP did away with many of OAuth 1.0's more confusing and problem-prone aspects, such as its custom signature calculation mechanism. After much debate in the community, WRAP was decided on as the basis for the new OAuth 2.0 protocol. Where OAuth 1.0 was monolithic, OAuth 2.0 was modular. The modularity in OAuth 2.0 allowed it to be a framework that could be deployed and used in all of the ways that OAuth 1.0 had been in practice, but without twisting core aspects of the protocol. OAuth 2.0 essentially provided recipes.

In 2012, the core OAuth 2.0 specifications were ratified by the IETF, but the community was far from done with it. This modularity was further codified by splitting the specification into two complementary pieces: RFC 6749 details how to get a token, while RFC 6750 details how to use a particular type of token (the Bearer token) at a protected resource. Furthermore, the core of RFC6749 details multiple ways to get a token and provides an extension mechanism. Instead of defining one complex method to fit different deployment models, OAuth 2.0 defines four different grant types, each suited to a different application type.

Today, OAuth 2.0 is the premier authorization protocol in use on the web. It's used by everything: from large internet companies to small startups, to enterprises, to just about everything in between and beyond. A whole ecosystem of extensions, profiles, and entire protocols built on top of OAuth 2.0 has sprung up, with people finding new and interesting ways to use this foundational technology. It's our goal that this book will help you understand not only what OAuth 2.0 is and why it works the way it does, but how you can best use it to solve your own problems and build your own systems.

JUSTIN RICHER

acknowledgments

Creating this book has been quite the journey. Ever since we embarked on the project and started putting the outline together, we had a feeling it was going to take a lot more sweat than we could have ever been prepared for. We were more right than we realized at the time, and it's with great pleasure that we are finally able to write this part, thanking the many people who helped make it happen. We can't possibly name you all here, so accept our humble thanks even if your name isn't listed here explicitly.

First off, this book would have never happened without the input and encouragement of the OAuth Working Group in the IETF and the larger OAuth and open standards communities. In particular, John Bradley and Hannes Tschofenig each provided invaluable input to the text at various points. Ian Glazer, William Dennis, Brian Campbell, Dick Hardt, Eve Maler, Mike Jones, and many others in the community encouraged us to create the book and helped provide important information to the internet. Aaron Parecki provided us space on oauth.net to not only talk about the book but also publish topical articles, including an early form of what became chapter 13. And special thanks to Ian for contributing the foreword and endorsing our work.

This book would literally not exist without the help and input from the team from Manning Publications. Our fantastic team of editors and support staff included Michael Stephens, Erin Twohey, Nicole Butterfield, Candace Gillhoolley, Karen Miller, Rebecca Rinehart, Ana Romac, and especially our amazing editor Jennifer Stout. Thanks to Ivan Kirkpatrick, Dennis Sellinger, and David Fombella Pombal for making sure the technical bits made sense. A big thanks to everyone who took a

chance and preordered the book as a MEAP; the early feedback we got from you was vital in making this the best book we could make it.

We would also like to thank our peer reviewers who read the manuscript at various stages of its development and provided invaluable feedback along the way: Alessandro Campeis, Darko Bozhinovski, Gianluigi Spagnuolo, Gregor Zurowski, John Guthrie, Jorge Bo, Richard Meinsen, Thomas O'Rourke, and Travis Nelson.

Justin Richer

Incomparable thanks are due to my coauthor, Antonio Sanso. His security and cryptographic expertise far outstrips anything I could dream of achieving, and it's been an honor to work with him. Starting the book was his idea in the first place, and the whole project has been a collaborative effort.

Thanks to my friends Mark Sherman and Dave Shepherd, both of whom successfully published tech books before I first set words to the page. Their existence served to remind me that there was a light at the end of the tunnel, and their experience in navigating the publishing world was a great help. Thanks to John Brooks, Tristan Lewis, and Steve Moore, whom I was able to bounce ideas and phrases off of, even if they didn't always realize I was doing it at the time.

Many thanks to my clients over the last year for putting up with me disappearing at random times to go off and write. Thanks are especially due to Debbie Bucci and Paul Grassi, as their fantastic work programs have helped give me the direct experience needed to ground this book in the real world.

I can't possibly express enough thanks to my friend and colleague, Sarah Squire. She originally turned me on to the Node.js frameworks used in the exercises throughout the book, and I believe that, thanks to a trip to an office store, she has the distinction of owning the first printed version of this book. Overall, her encouragement, support, critique, and enthusiasm for this project has been without compare, and I doubt that the book would have really happened without her.

Finally, but perhaps most importantly, a sincere and deep thank you to my entire family. The patience of my wife, Debbie, and my kids, Lucien, Genevieve, and Xavier, has been incredible. Between late nights and seemingly endless weekends with me locked up in my office, just out of reach, I'm sure they started to wonder if I'd ever come out, but now I'm glad to say there should be a whole lot more time to play Legos.

Antonio Sanso

Working on this book has been quite a ride, and it's with great delight and satisfaction that I write this part. In the end, as with everything, it's the journey and not the destination that matters. My contribution to this book could not be possible without the help of many people surrounding me.

I would like to thank my employer, Adobe Systems, and my managers Michael Marth and Philipp Suter for giving me the green light to work on this book.

OAuth is a widespread protocol written in a collaborative way by many people under the IETF umbrella. Some of those people are the brightest minds in the security community. We had the privilege to have some extremely useful comments on the work-in-progress draft by John Bradley, Hannes Tschofenig and William Denniss.

It is incredible how friendship can have an influence on someone's life. For this reason, I'd like to thank, in no particular order: Elia Florio for being a constant source of inspiration; Damien Antipa for being so patient while explaining the most arcane part of Javascript and CSS; Francesco Mari, who introduced me to the beautiful world of Node.js and tirelessly listened my endless complains; Joel Richard for helping me with the magic of Apache Cordova; Alexis Tessier, the most talented designer I ever met; and Ian Boston for proofreading.

And last but not least, Justin Richer, who has been the best coauthor I could ever hope for. You rock, Justin!

But I can't finish without a special thank you to the people I love.

To my parents. They always encouraged me to pursue studying, without putting any pressure on me, even if they didn't study themselves. Their support was unique. To my brother and sister who also encouraged me, especially in the early stage of my university time.

And of course, the biggest thank you goes to my fiancée (soon wife) Yolanda, who supports and continuously encourages me on everything I do. Finally, to Santiago, my son, who helps me remember every single day how beautiful life is. I love you.

about this book

This book is intended to be a comprehensive and thorough treatment of the OAuth 2.0 protocol and many of its surrounding technologies, including OpenID Connect and JOSE/JWT. We want you to come away from this book with a deep understanding of what OAuth can do, why it works the way that it does, and how to deploy it properly and securely in an unsafe internet.

The target reader for this book is someone who's probably used OAuth 2.0, or at least heard of it, but doesn't really know how it works or why it works that way. Maybe you've even developed one or more OAuth 2.0 components, such as a client to talk to a specific API, but you're curious about other kinds of clients, or other parts of the OAuth 2.0 ecosystem. Perhaps you wonder, "What's the authorization server doing when you go ask for that authorization code, anyway?" Or perhaps you're tasked with protecting an API and you want to know if OAuth 2.0 is really going to do the job, and if so, how are you supposed to manage that? Maybe in your day job you're building a client, but you want to know what the protected resource does with that token you sent it. Or maybe you're building and protecting an API, but you want to know what the authorization server you're talking to does to get those tokens into the right place. We want you to understand what the tool, OAuth 2.0, is really good at and how you can wield it effectively.

We're going to assume you know the basics of how HTTP works, and at least understand the utility of encrypting connections using TLS, if not the intimate details of how it works. Our code is all in JavaScript, but this isn't a book about JavaScript, and so we've done our best to explain the abstractions and functionality that the code itself represents so that you can apply it to your own platform and language.

Roadmap

This book has 4 sections consisting of 16 chapters in total. The first section, consisting of chapters 1 and 2, provides an overview of the OAuth 2.0 protocol and is considered core reading material. The second section, consisting of chapters 3 to 6, demonstrates how to build an entire OAuth 2.0 ecosystem. The third section, consisting of chapters 7 to 10, discusses vulnerabilities to different parts of the OAuth 2.0 ecosystem and how to avoid them. The final section, consisting of chapters 11 to 16, goes beyond the core OAuth 2.0 protocol and into the wider ecosystem of standards and specifications, as well as providing a wrap-up to the book.

- Chapter 1 provides an overview of the OAuth 2.0 protocol, as well as the motivation behind its development, including approaches to API security that predates OAuth.
- Chapter 2 goes into depth on the authorization code grant type, the most common and canonical of OAuth 2.0's core grant types.
- Chapters 3 through 5 demonstrate how to build a simple but fully functional OAuth 2.0 client, protected resource server, and authorization server (respectively).
- Chapter 6 looks at the variations in the OAuth 2.0 protocol, including grant types other than the authorization code, as well as considerations for native applications.
- Chapters 7 through 9 discuss common vulnerabilities in OAuth 2.0 clients, protected resources, and authorization servers (respectively) and how to prevent them.
- Chapter 10 discusses vulnerabilities and attacks against OAuth 2.0 bearer tokens and authorization codes and how to prevent them.
- Chapter 11 looks at JSON Web Tokens (JWT) and the JOSE mechanisms used in encoding them, as well as token introspection and revocation to complete the token lifecycle.
- Chapter 12 looks at dynamic client registration and how that affects the characteristics of an OAuth 2.0 ecosystem.
- Chapter 13 looks at how OAuth 2.0 is not an authentication protocol, and then proceeds to show how to build an authentication protocol on top of it using OpenID Connect.
- Chapter 14 looks at the User Managed Access (UMA) protocol built on top of OAuth 2.0 that allows for user-to-user sharing, as well as the HEART and iGov profiles of OAuth 2.0 and OpenID Connect and how these protocols are applied in specific industry verticals.
- Chapter 15 moves beyond the common bearer token of OAuth 2.0's core specifications and describes how both Proof of Possession (PoP) tokens and TLS token binding work with OAuth 2.0.

- Chapter 16 wraps everything up and directs the reader to how to apply this knowledge going forward, including a discussion of libraries and the wider OAuth 2.0 community.

We don't expect you to read this book in order, though you can do just that and we've tried to organize things to allow that kind of exposition. We do suggest that you read the first two chapters together, because they'll give you a very thorough overview of OAuth 2.0 and provide some deep looks into key concepts and components. But let's be honest, you're probably looking for specific bits of information, so maybe you'll go read the chapters on client development and client vulnerabilities, then hop around to the chapter on user authentication or token management, and then go take a look at how authorization servers tick. Because of this, we've also tried to make sure that each chapter really stands on its own, and we've put in references for other content throughout the book so that you can find your way to topics.

About the code

All of the code in this book is available as open source under an Apache 2.0 license. We feel that it's important to encourage people to use, remix, and contribute to the code, even if they're just exercises and examples. The worlds of open standards, like OAuth, and open source go hand in hand, and we feel it's important that we help contribute to that. The source is available from GitHub at https://github.com/oauthinaction/oauth-in-action-code/ and we encourage you to fork it, clone it, branch it, and even make pull requests to make it better. Code exercises are available for chapters 3 to 13, and 15, with an overview of the framework available in appendix A and selected code listings in appendix B. The code is also available for download from the publisher's website at www.manning.com/books/oauth-2-in-action.

All of the code in this book is written in the JavaScript language using Node.js. Web applications, which comprise most of the examples, use Express.js and a variety of other libraries to function. We've tried our best to insulate the readers from the oddities of JavaScript, as the goal of this book is not to learn proficiency in a particular language or platform. If you've ever programmed with a web framework, such as Java Spring or Ruby on Rails, then you'll be familiar with most of the concepts and constructs. Furthermore, we've tried to include documented utility functions to handle some of the ancillary details to the OAuth protocol, such as building a properly formatted and encoded URL with query parameters or creating an HTTP Basic authentication string. See appendix A for more details on the code environment used throughout the book, including a simple exercise designed to show the reader how to get things up and running.

Selected exercises are also available online at Katacoda (www.katacoda.com), an interactive, self-guided tutorial website. These exercises use the exact same code as the book itself, but are presented in a containerized runtime environment available over the web.

Code conventions

This book contains many examples of source code both in numbered listings and in line with normal text. In both cases, source code is formatted in a `fixed-width font like this` to separate it from ordinary text. Sometimes code is also in **`bold`** to highlight code that has changed from previous steps in the chapter, such as when a new feature adds to an existing line of code.

In many cases, the original source code has been reformatted; we've added line breaks and reworked indentation to accommodate the available page space in the book. In rare cases, even this wasn't enough, and listings include line-continuation markers (➥). Additionally, comments in the source code have often been removed from the listings when the code is described in the text. Code annotations accompany many of the listings, highlighting important concepts.

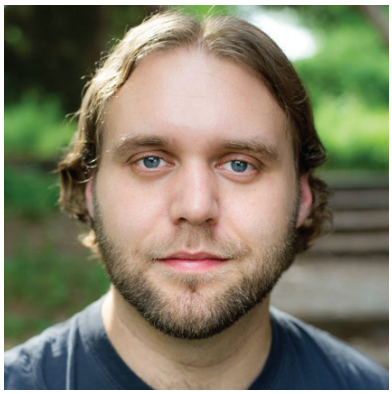
Author Online

The purchase of *OAuth 2 in Action* includes free access to a private web forum run by Manning Publications, where you can make comments about the book, ask technical questions, and receive help from the authors and from other users. To access the forum and subscribe to it, point your web browser to www.manning.com/books/oauth-2-in-action. This page provides information on how to get on the forum once you are registered, what kind of help is available, and the rules of conduct on the forum.

Manning's commitment to our readers is to provide a venue where a meaningful dialogue between individual readers and between readers and the authors can take place. It is not a commitment to any specific amount of participation on the part of the authors whose contribution to the forum remains voluntary (and unpaid). We suggest you try asking the authors some challenging questions lest their interest stray!

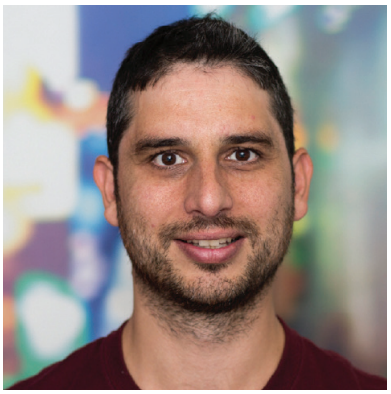
The Author Online forum and the archives of previous discussions will be accessible from the publisher's website as long as the book is in print.

about the authors



Justin Richer is a systems architect, software engineer, standards editor, and service designer with over seventeen years of industry experience in a wide variety of domains including internet security, identity, collaboration, usability, and serious games. As an active member of the Internet Engineering Task Force (IETF) and OpenID Foundation (OIDF) he has directly contributed to a number of foundational security protocols including OAuth 2.0 and OpenID Connect 1.0, as well as being the editor of several extensions of OAuth 2.0 including Dynamic Client Registration (RFC7591 & RFC7592) and Token Introspection (RFC7662). His pioneering work with Vectors of Trust and the third edition of NIST's Digital Identity Guidelines (Special Publication 800-63) have pushed the conversation of what a trusted identity means in an unpredictable landscape. He is the founder and maintainer of the enterprise-focused MITREid Connect open source implementation of OAuth 2.0 and OpenID Connect and has led production deployment of the system at a number of organizations including The MITRE Corporation and the Massachusetts Institute of Technology. An accomplished and confident presenter, he is much sought-after as a plenary and keynote speaker at conferences around the world to audiences of all technical proficiencies. An ardent proponent

of open standards and open source, he believes in solving hard problems with the right solution, even if that solution still needs to be invented.



Antonio Sanso works as Senior Software Engineer at Adobe Research, Switzerland, where he is part of the Adobe Experience Manager security team. Prior to this, he worked as software engineer in the IBM Dublin Software Lab in Ireland. He found vulnerabilities in popular software, such as OpenSSL, Google Chrome, and Apple Safari, and he is included in the Google, Facebook, Microsoft, Paypal, and Github security hall of fame. He is an avid open source contributor, being the Vice President (chair) for Apache Oltu and a PMC member for Apache Sling. His working interests range from web application security to cryptography. Antonio is also the author of more than a dozen computer security patents and applied cryptography academic papers. He holds an MSc in Computer Science.

about the cover illustration

The figure on the cover of *OAuth 2 in Action* is captioned "Man from Zagrovic, Dalmatia, Croatia." The illustration is taken from a reproduction of a mid-nineteenth century album of Croatian traditional costumes by Nikola Arsenovic, published by the Ethnographic Museum in Split, Croatia, in 2003. The illustrations were obtained from a helpful librarian at the Ethnographic Museum in Split, which is located within the ruins of Emperor Diocletian's retirement palace from around AD 304, in the Roman core of the medieval center of the town. The book includes finely colored illustrations of figures from different regions of Croatia, accompanied by descriptions of the costumes and of everyday life.

Zagrovic is a small town in inland Dalmatia, built on the ruins of an old medieval fortress. The figure on the cover is wearing blue woolen trousers and, over a white linen shirt, a voluminous red woolen jacket, richly trimmed with the colorful embroidery typical for this region. He is holding a long pipe in one hand and has a musket slung over his other shoulder. A red cap and leather moccasins complete the outfit.

Dress codes and lifestyles have changed over the last 200 years, and the diversity by region, so rich at the time, has faded away. It is now hard to tell apart the inhabitants of different continents, let alone of different hamlets or towns separated by only a few miles. Perhaps we have traded cultural diversity for a more varied personal life—certainly for a more varied and fast-paced technological life.

Manning celebrates the inventiveness and initiative of the computer business with book covers based on the rich diversity of regional life of two centuries ago, brought back to life by illustrations from old books and collections like this one.